AUDIOVISUAL AND PEDAGOGICAL NETWORK INSTALLATION

Ludovic Laffineur

LARAS - IRISIB
150 Rue Royale,
1000 Brussels, Belgium
ludovic.laffineur@irisib.be

Rudi Giot

LARAS - IRISIB
150 Rue Royale,
1000 Brussels, Belgium
giot@isib.be

Louis Commère

PHELMA
3 Parvis Louis Néel,
38000 Grenoble, France
lcommere@gmail.com

ABSTRACT

This paper presents an interactive installation designed to inform visitors of the network flow and the risks to connect their devices to any Wi-Fi hotspots. The system developed in C++ grabs packets using LibPCap, analyses them at low level (e.g., packet length) and also provides high-level information (e.g., port number). This new approach is based on the network flow analysis as well as on network services analysis. The software communicates with ChucK through the OSC protocol and is developed with the OpenFrameworks library in order to create unique visualisation. Users can actively take part to an interactive and didactic audiovisual exhibition system using their mobile device to send e-mails, listen to a web radio, surf on a website, read RSS feeds, in short, the experience begins once visitors exchange data with the network.

1. INTRODUCTION

Networked systems are now ubiquitous in the modern world. The growing number of hotspots and services, allows us to be connected permanently. However, an important drawback is that networks become increasingly vulnerable to hacking or usage abuses.

In order to control those misuses, network administrators monitor the infrastructure and analyse people's behaviours[1]. A typical weakness in network is non-encrypted authentication. Connected computers can know exactly the content of a non-encrypted communication and administrators could trace users' activity. The deontology forbids administrators from interfering with users' privacy depending on the type of network (i.e. private, professional or public network) but malicious people would readily take advantage of such a security vulnerability (i.e. non-encrypted authentication process).

Behind every click on a website, hidden requests are launched to get services (e.g., Facebook share buttons), advertisements, or to post website statistics (e.g., Google Analytics), thus tracing the visitors' actions. Most people are concerned with personal data protection on social networks but few are conscious of the concealed features of most websites.

The current paper presents a sonification and visualisation installation that should increase people awareness in network abuses, vulnerabilities of authentication processes and hidden website queries they might undergo. Visitors and their children can discover network properties by interacting with the installation. For example IP addresses are displayed on different locations on a map, and different service types and lower levels properties (such as bandwidth or payload) are represented by different synthesizers sounds. Once users are connected to the Wi-Fi hotspot provided by the computer, the system analyses packets and renders visitors' actions.

2. EXISTING PROJECTS

This section describes several previous projects related to network sonification. Most of them are not dealing with a pedagogical aspect, unlike this paper.

The PEEP [2] approach is to eliminate the need to search through large amounts of text by rendering network information in real-time. A network administrator could tell the current activity on the network. Mail, DNS queries, telnet traffic, load average and number of users are events handled by PEEP. Sounds provided to the listener an approximation of network state.

C.Chafe and A.Leistikow [3] created auditory tools for evaluating network performance. Authors used the ping utility to get a momentary measurement of round trip time. Sequences of ping events are stored to gather statistics to describe the quality of service of a network path.

Net Sonification [4] was a program written in JAVA that crawls

 The present work is supported by the Research Laboratory in the field of Arts and Sciences, LARAS (Brussels) and "La Région Wallonne" through the ACTION project which is a "FIRST Haute-école" program from the "DGO6 Département des Programmes de Recherche".

across the internet, grabbing as many related URLs as possible and analysing their content. Using Max/MSP [5], the data coming from the web-crawler program is translated into sound. On the project's website, a series of mp3s are offered which are basically sound portraits of a particular website's content and related link structure. This project enabled the listener to experience the sound and rhythm of the network as a time based medium, rather than visually as discrete semantic pages.

Toward Sound-Assisted Intrusion Detection Systems [6]. Their system detected intrusions and attacks on a network. The intrusion detection system effectively generated distinctive sounds upon a series of simple attack scenarios consisting of denial-of-service and port scanning.

Leech: BitTorrent and music piracy sonification [7]. Leech could acoustically and visually render BitTorrent [8] traffic. The nature and usage of BitTorrent networking is discussed, including the implications of wide-spread music piracy.

SonNet[9] executed packet-sniffing and network connection state analysis automatically, and a ChucK[10] object is developed and used from a user's own code. The system is designed to make computer network data easily accessible for composers.

WireShark project [11] is a free and open-source packet analyser based on LibPcap library and is the reference in network troubleshooting and analysis. Unfortunately deep knowledges in network are required in order to extract understandable data from this software.

These projects introduced different aims of the network rendering. PEEP is written in the beginning of the Internet expansion and authors already felt the lack of gathering instant and continuous information to monitor the network. Leech introduced the visual link between IP addresses and and location but is limited to BitTorrent traffic. SonNet makes network properties available for composers and is a step to learn network properties and implications. Although, the stand-alone software WireShark only purposes to analyse network flow and can not interact with any application.

Other papers introduce the sonification of hacking attempts and network performances.

3. REQUIREMENTS

The aim is to create an installation where people just have to connect their devices to a Wi-Fi hotspot and instantly perceive their actions on the network.

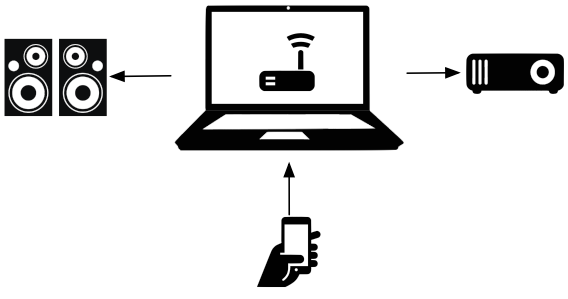

Figure 1: Wi-Fi hotspot sonification: The computer creates Wi-Fi hotspot, visitors connect their mobile devices to the Wi-Fi hotspot and the system produces sounds and projections.

The computer of the installation shares its Ethernet connection through Wi-Fi (Figure 1) and the software plays the behaviour of other people connected to the network. Therefore the minimalist version of the application only requires an Internet connection (cable) and power supply. Optionally speakers or additional projectors can be added to create a complete audiovisual environment.

4. DATA GRABBING AND MAPPING

In this section the packet capture module and the mapping between packets parameters and audiovisuals properties are explained. Visitors send datagrams with their mobile devices on the Wi-Fi. The system analyses packets and sends messages to the Text-To-Speech (TTS) system of MacOSX and Open Sound Control [12] (OSC) messages to ChucK and to an OpenFrameworks [13] application which generates visual projections. (Figure 2).

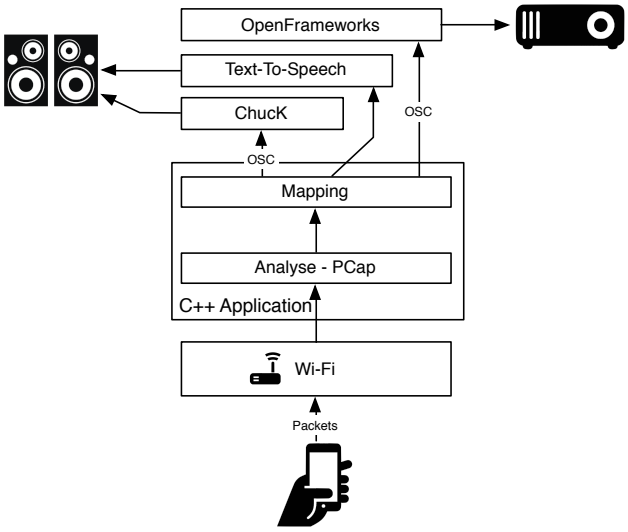

Figure 2: Visitors connects their device to the Wi-Fi hotspot. Datagrams are grabbed using the PCap Library and rendered with OpenFrameworks and ChucK

4.1. Packet Capture

The C++ PCap library (WinPCap, LibPCap) [14] is used in order to analyse the traffic at very low level (OSI model's layer 3 and lower [15]) and also to get some high level information. Complementary information about IP packets can be found elsewhere (see [16]). Concerning this application, parameters extracted from the packets are

- the packet length that indicates the weight of the packet,
- IP addresses to know the origin of the transmission,
- the port number to know which service is called (e.g., HTTP, FTP),
- the number of packets going through the network in a given amount of time is used to evaluate the saturation level of the network,
- Dynamic Host Configuration Protocol (DHCP) requests to catch visitors connections to the Wi-Fi,

- non-encrypted passwords with a regular expression applied to the packet's content and,
- Domain Name Service (DNS) requests to grab the name of visitors' requested websites.

The figure 3 shows how parameters are mapped with ChucK, OpenFramework and MacOSX. Following sections will describe in detail how to extract intelligible data in such low level parameters and reasons of using ChucK and OpenFrameworks.

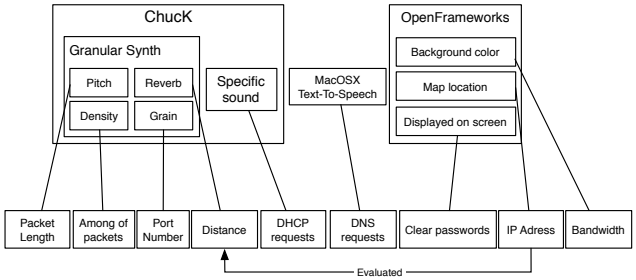

Figure 3: The mapping of packet parameters with properties of audiovisual software.

4.2. Distance Evaluation

Public IP addresses in the world are distributed in pools corresponding to specific regions and are available in a free database GeoLite[17]. This database provides the country of a range of IP addresses but requires some adaptations that are made in a Python [18] pre-process script at the initialisation time. In fact, countries are converted to GPS locations using the Google Maps API. Knowing GPS positions of two points the distance between them is calculated using the haversine formula [19]:

$$\begin{aligned} e &= \sqrt{\sin^2\left(\frac{\phi_2-\phi_1}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\lambda_2-\lambda_1}{2}\right)} \\ d &= 2r\arcsin(e) \end{aligned} \tag{1}$$

where :

- d is the distance between the two points,
- r is the radius of the sphere,
- ϕ_1, ϕ_2: latitude of point 1 and 2,
- λ_1, λ_2: longitude of point 1 and 2.

Dividing d by maximum distance between two point on earth (i.e., the half equatorial distance, 20 000 km), we get a number between $[0, 1]$ that we add in the database as the distance ratio.

$$\text{Ratio} = \frac{d}{20000\text{ km}} \tag{2}$$

4.3. Research in the Database

IP addresses are composed of 4 bytes and are stored as strings and integer in the database. String IP address (e.g., 192.168.1.100) can be converted to an integer with this formula:

$$W.X.Y.Z = 2^{24} * W + 2^{16} * X + 2^8 * Y + Z \tag{3}$$

The IP address field extracted from the packet is an integer and can be straightly compared to integers values in the database. A new table is created only with fields used by the application and its structure is composed of 4 fields (figure 4): the beginning IP address of the pool, the latitude, the longitude and the distance ratio. Note that latitudes and longitudes are in radians.

begin_ip_address	latitude	longitude	distance_ratio
16777216	-0.4615	2.3262	0.745
16777472	0.6259	1.8176	0.382
...	...	...	...

Figure 4: Structure of the modified GeoLite table: Latitudes and longitudes are in radians. In this example, the first row corresponds to Australia and the second one to China.

GeoLite table contains 84000 entries and each grabbed packet searches its corresponding country in the table. In order to improve the search speed, the modified table is loaded in main memory ordered by growing IP addresses to avoid disk accesses and the location is found using the binary search algorithm [20]. The pseudo-code can be found in figure 5.

```
BinarySearch(A[0..N-1], value) {
  low = 0
  high = N - 1
  while (low <= high) {
    mid = (low + high) / 2
    if (A[mid] > value)
      high = mid - 1
    else if (A[mid] < value)
      low = mid + 1
    else
      return mid
  }
  return not_found
}
```

Figure 5: Pseudo-code of the binary search algorithm: each iteration reduces the possibilities by half.

Each iteration reduces the interval by two and thus the iteration count to cover all possibilities is calculated with:

$$\begin{aligned} I_{max} &= \lceil \log_2 84000 \rceil \\ I_{max} &= 17 \end{aligned} \tag{4}$$

In a maximum of 17 iterations, the algorithm parses 84000 countries and converges to the solution, satisfying even the strictest of speed requirements.

4.4. ChucK

"ChucK is a programming language for real-time sound synthesis and music creation" [10]. A lot of synthesis types such as a granular synthesis can be implemented very conveniently. Although ChucK is able to run many processes (i.e. shreds) in parallel and support MIDI and OSC protocols. As illustrated in Figure 2, the application controls ChucK using OSC messages.

The granular synthesiser has four controlled parameters:

- the density of grains. Theoretically every grain corresponds to a packet that renders his own sound and the among of packets is modelled in practise by the density of grains.
- the type of grains. Depending on the port number, grains' timber change to display auditorily the difference between HTTP or FTP protocols for instance.
- the reverberation. This parameter is linked to the packet's origin (i.e. distance ratio).
- the pitch of grains is commanded by the packet length parameter. A heavy packet will make a low frequency and a light packet a high frequency.

Moreover ChucK plays a sound when a new device connects to the Wi-Fi hotspot (DHCP request).

4.5. OpenFrameworks

OpenFrameworks is free and open source C++ toolkit providing an intuitive framework for experimentation. Lots of common used libraries are wrapped in this framework such as OpenGL [21] for graphics or OpenCV [22] for computer vision. The visualisation of network is a map with located points (remote servers) linked to the current location with a weight. Grabbed passwords are displayed in the bottom of screen (see figure 6).

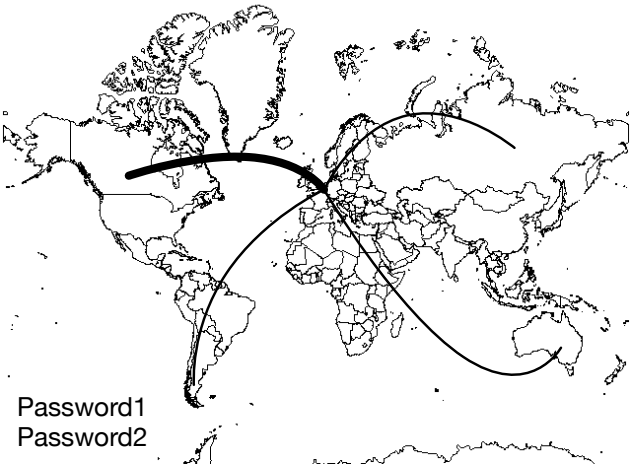

Figure 6: Visualisation of the network flow: located points are linked with a weight to the current location and non-encrypted password are displayed.

5. CONCLUSION

The system is an innovative network sonification software that can be applied in many places because necessary hardware prerequisites are low. The application offers a new way to learn basic network parameters and the related risks that users have to face.

In the beginning of 2014 a final prototype version will be set up in a public place in Brussels (Point Culture) for public tests. These tests will be driven into two directions: user interactions and software reliability.

6. ACKNOWLEDGMENT

We thank Alexandra Degeest (Brussels, BE), Christian Frisson (Université de Mons numédiart, BE) and Vincent Berthiaume (University of Montreal, CA) for useful comments on an earlier version of this manuscript.

7. REFERENCES

[1] "Nac - network analyser with c#," http://nac.dev.isib.be.

[2] M. Gilfix and A. L. Couch, "Peep (the network auralizer): Monitoring your network with sound," in *Proceedings of the 14th USENIX Conference on System Administration*, ser. LISA '00. Berkeley, CA, USA: USENIX Association, 2000, pp. 109–118.

[3] C. Chafe and A. Leistikow, "Levels of temporal resolution in sonification of network performance," in *Proceedings of the 2001 International Conference on Auditory Display*, Espoo (FI), 2001, pp. 50–55.

[4] Z. Layton, "Network sonification," http://turbulence.org/Works/net_sonification/.

[5] "Max/msp," http://cycling74.com/.

[6] L. Qi, M. V. Martin, B. Kapralos, M. Green, and M. A. Garcia-Ruiz, "Toward sound-assisted intrusion detection systems." in *OTM Conferences (2)*, ser. Lecture Notes in Computer Science, R. Meersman and Z. Tari, Eds., vol. 4804. Springer, 2007, pp. 1634–1645.

[7] C. McKinney and A. Renaud, "Leech: Bittorrent and music piracy sonification," *Sound and Music Computing*, 2011.

[8] "Bittorrent protocol specification," http://bittorrent.org/beps/bep0003.html.

[9] K. E. Wolf and R. Fiebrink, "Sonnet: A code interface for sonifying computer network data," in *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, South Korea, 2013, pp. 503–506.

[10] G. Wang, "The chuck audio programming language: A strongly-timed and on-the-fly environ/mentality," Ph.D. dissertation, Princeton University, 2008.

[11] "Wireshark, go deep," http://www.wireshark.org/.

[12] "Open sound control," http://opensoundcontrol.org/.

[13] "openframeworks," http://openframeworks.cc/.

[14] "Libpcap," http://www.tcpdump.org/.

[15] "Microsoft, the osi model's seven layers defined and functions explained," http://support.microsoft.com/kb/103884.

[16] "Tcp / ip reference page," http://www.protocols.com/pbook/tcpip1.htm.

[17] "Geolite," http://dev.maxmind.com/geoip/legacy/geolite/.

[18] "Python programming language – official website," http://python.org.

[19] "Haversine formula," http://en.wikipedia.org/wiki/Haversine_formula.

[20] R. Panneerselvam, *Design and Analysis of Algorithms*. New-Delhi: Prentice-Hall of India Pvt.Ltd, 2007, ch. 5, pp. 118–184.

[21] "Open source graphic library," http://www.opengl.org/.

[22] "Open source computer vision," http://opencv.org/.